An Empirical Evaluation of a Combinatorial Auction for Solving Multi-Agent Pathfinding Problems

An Empirical Evaluation of a Combinatorial Auction for Solving Multi-Agent Pathfinding Problems

Ofra Amir
Guni Sharon
and
Roni Stern

TR-05-14

Computer Science Group
Harvard University
Cambridge, Massachusetts

An Empirical Evaluation of a Combinatorial Auction for Solving Multi-Agent Pathfinding Problems

Ofra Amir[1], Guni Sharon [2], and Roni Stern [2]

[1]School of Engineering and Applied Sciences, Harvard University
[2]Department of Information Systems Engineering, Ben-Gurion University of the Negev

1 Introduction

This technical report provides an empirical evaluation of the implementation of iBundle for solving optimal MAPF problems as proposed by the authors in [1]. It also includes and a detailed running example of iBundle on an MAPF problem.

2 Empirical Evaluation

To demonstrate the applicability of our general approach for solving MAPF problems [1], which is based on a reduction to CA and using appropriate CA mechanism, we evaluated experimentally our implementation of iBundle for solving optimal MAPF problems (denoted simply as iBundle). iBundle was compared to the Increasing Cost Tree Search (ICTS) [4] and Conflict Based Search (CBS) [2, 3], two recent MAPF algorithms. In general, we found that iBundle was competitive with these solvers. We also ran A^* on all problems, but we do not report these results as all other algorithms significantly outperformed it.

ICTS and CBS were run with their recommended configurations [2, 3]. All algorithms were run within the Independence-Detection (ID) framework [5]. This framework decomposes the problem to subproblems including subgroups of the agents that can be solved independently, leading to a lower effective problem size. For all algorithms, if no solution was found within 5 minute we consider it as a failure.

The top rows of Table 1 show the runtime of the algorithms when solving problems on 3×3 grids, with the number of agents ranging between 4 and 8. For each number of agents, we randomly generated 50 instances and averaged their results. The k' column shows the average effective number of agents after ID (i.e., the average number of agents in each sub-problem). Runtime results are shown only for instances that were successfully solved by all algorithm. The three algorithms were always successful at finding solutions to problems with up to 7 agents in the allotted time. With 8 agents, CBS succeeded in solving only 44% of the instances while iBundle succeeded in solving 75% of the instances and ICTS solved 81% of the instances successfully. As can be seen in the table, ICTS performs best, followed by iBundle and CBS.[1] Overall, iBundle scales well with the number of agents and outperforms CBS in these grids. Results for 8×8 grids (bottom lines of table 1) show a similar trend.

[1]Earlier work [2] reported that CBS outperformed ICTS; the differences are likely due to changes in the implementation of the algorithms and their overall performance remains similar.

k	k′	iBundle	ICTS	CBS
3 × 3				
4	2.1	10	8.14	21.5
5	2.6	14.1	7.8	15.5
6	4.7	169.1	50.9	2,314.1
7	6.2	2,359.6	463.2	17,238.5
8	7.8	18,322.4	4,093.5	67,380.7
8 × 8				
8	1.7	18.4	15.3	19.1
9	2	29.1	17	35.3
10	2.3	243.3	22.4	32.8
11	2.4	45.2	27.1	114.3
12	3.4	265.2	40.7	135.4
13	3.7	305.6	44.6	357.9
14	4.8	2,476.2	231.5	7,810.6

Table 1: Runtime (ms) results on 3×3 and 8×8 grids with no obstacles.

We also ran the algorithms on two maps, den520 and brc202d, from the "Dragon Age" video game, taken from Sturtevant's repository [6]. We varied the number of agents between 5 and 30. Table 2 shows the runtime of the algorithms on these maps (averaged over 50 randomly generated problem instances). When solving den520 instances, iBundle outperformed CBS but was slower than ICTS (similar to the trend seen in the smaller grids). For maps with 30 agents, ICTS successfully solved 98% of the problems, followed by iBundle with 94% and CBS with 80%. When solving brc202d instances, iBundle was outperformed by both ICTS and CBS for most sizes of agents. iBundle was successfully solved 82% of the instances with 30 agents compared to ICTS with 94% and CBS with 100%.

These empirical results demonstrate that in addition to being capable of addressing MAPF variants that are not addressed by state-of-the-art MAPF algorithms (e.g., MAPF for self-interested agents), our auction approach is also competitive with state-of-the-art optimal MAPF algorithms in the cooperative settings and can be used in practice.

3 iBundle Example

Next, we describe in details how the iBundle mechanism works on the MAPF problem shown in Figure 1. Table 3 shows the agents' bids, the auctioneer's revenue, the provisional allocation and the new prices according to the iBundle auction process for the MAPF problem depicted in Figure 1. In the first round the ask price for all paths is 0, and the agents place bids on their shortest paths (shown for each agent in the table). Since there is no feasible allocation that includes all agents, the provisional allocation includes only two agents, a_1 and a_3. Since no paths were allocated to a_2, the prices of all the paths it bid on are raised to 1 (shown in the rightmost column).

In the second round, a_1 and a_3 maintain the same bids, as their prices remain 0. Agent a_3 continues to bid on its shortest paths (with path cost 4) that now have a price of 1 and also bids on all of its paths of cost 5 at a price of 0. Again, there is no feasible allocation of all agents and the allocation that maximizes the auctioneer's revenue only includes a_2 which is now willing to pay 1 for its shortest paths. No other agent is included in the allocation as there is no non-conflicting allocation given the current bids of a_1 and a_3.

k	k′	iBundle	ICTS	CBS
den520d				
5	1	262.4	41.4	72.7
10	1.1	354.9	126.0	281.5
15	1.3	1263.9	319.2	5504.6
20	1.4	1088.6	298.4	5853.7
25	1.5	4265.3	920.3	3880.0
30	1.8	1882.2	602.3	7926.6
brc202d				
5	1.1	195.6	95.2	243.4
10	1.3	999.9	320.8	648.0
15	1.6	7368.2	2961.5	1474.3
20	1.7	1403.3	636.8	4250.6
25	2	14657.3	1832.2	3561.1
30	2.5	20226.1	2380.0	4157.2

Table 2: Runtime (msec) of the algorithms on the "den520d" and "brc202d" dragon age maps (257×256 with 37614 obstacles and 481×530 with 211779 obstacles respectively).

	a_1	a_2	a_3	Provisional Allocation	Revenue	New prices
Round 1	{AC,0}	{BACE,0};	{FDCEI, 0};	A1: {AC,0};A3: {FDGEI,0}	0	{BACE,1};
		{BDCE,0};	{FDGEI, 0};			{BDCE,1};
		{BDGE,0}				{BDGE,1}
Round 2	{AC,0}	{BACE,1};	{FDCEI, 0};	A2: {BACE,1}	1	{AC,1};
		{BDCE,1};	{FDGEI, 0}			{FDCEI, 1};
		{BDGE,1};				{FDGEI, 1}
		{BBACE,0};				
		{BBDCE,0};				
		{BBDGE,0};				
		...				
Round 3	{AC,1};	{BACE,1};	{FDCEI, 1};	A1: {AC,1};A2: {BBDGE,0}; A3: {FDGEI,1}	2	The auction terminates
	{AAC,0};	{BDCE,1};	{FDGEI, 1};			
	{ACC,0}	{BDGE,1};	{FFDCEI, 0};			
		{BBACE,0};	{FFDGEI, 0};			
		{BBDCE,0};	...			
		{BBDGE,0};				
		...				

Table 3: An example of the iBundle auction process on the problem shown in Figure 1.

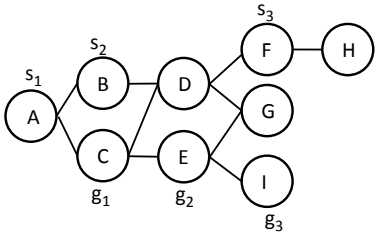

Figure 1: An example MAPF problem with three agents, a_1, a_2 and a_3. s_i and g_i mark the start and goal locations for and agent i.

Following this allocation, the prices for the paths a_1 and a_3 bid on increase to 1 as they were not included in the provisional allocation.

Next, in the third round, a_1 and a_3 bid on their shortest paths at a price of 1, and in addition bid on their next best paths at a price of 0. a_2 maintains its bids as the prices for its paths did not change. Finally, the revenue maximizing allocation includes all agents and the auction terminates with the optimal path allocation.

References

[1] Ofra Amir, Guni Sharon, and Roni Stern. Multi-agent pathfinding as a combinatorial auction. In *AAAI*, 2015.

[2] Guni Sharon, Roni Stern, Ariel Felner, and Nathan R Sturtevant. Conflict-based search for optimal multi-agent path finding. In *AAAI*, 2012.

[3] Guni Sharon, Roni Stern, Ariel Felner, and Nathan R Sturtevant. Meta-agent conflict-based search for optimal multi-agent path finding. In *SOCS*, 2012.

[4] Guni Sharon, Roni Stern, Meir Goldenberg, and Ariel Felner. The increasing cost tree search for optimal multi-agent pathfinding. *Artificial Intelligence*, 2012.

[5] Trevor Scott Standley. Finding optimal solutions to cooperative pathfinding problems. In *AAAI*, 2010.

[6] N. Sturtevant. Benchmarks for grid-based pathfinding. *Transactions on Computational Intelligence and AI in Games*, 4(2):144 – 148, 2012.